



UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

| APPLICATION NO. | FILING DATE | FIRST NAMED INVENTOR | ATTORNEY DOCKET NO. | CONFIRMATION NO. |
|-----------------|-------------|----------------------|---------------------|------------------|
| 09/731,617 | 12/07/2000 | James Allan Kahle | AT9-99-445 | 1084 |

7590

03/08/2004

Joseph P. Lally
DEWAN & LALLY, L.L.P.
P.O. Box 684749
Austin, TX 78768-4749

| |
|----------|
| EXAMINER |
|----------|

GOLE, AMOL V

| | |
|----------|--------------|
| ART UNIT | PAPER NUMBER |
|----------|--------------|

2183

DATE MAILED: 03/08/2004

Please find below and/or attached an Office communication concerning this application or proceeding.

27

Office Action Summary

Application No.

09/731,617

Applicant(s)

KAHLE ET AL.

Examiner

Amol V. Gole

Art Unit

2183

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --
Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

- 1) ☒ Responsive to communication(s) filed on 20 February 2004.
- 2a) ☒ This action is **FINAL**. 2b) ☐ This action is non-final.
- 3) ☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

- 4) ☐ Claim(s) _____ is/are pending in the application.
- 4a) Of the above claim(s) _____ is/are withdrawn from consideration.
- 5) ☐ Claim(s) _____ is/are allowed.
- 6) ☒ Claim(s) 1-4, 6-13 and 20-22 is/are rejected.
- 7) ☐ Claim(s) _____ is/are objected to.
- 8) ☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

- 9) ☒ The specification is objected to by the Examiner.
- 10) ☒ The drawing(s) filed on 20 February 2004 is/are: a) ☒ accepted or b) ☐ objected to by the Examiner.
Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).
Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).
- 11) ☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

- 12) ☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).
- a) ☐ All b) ☐ Some * c) ☐ None of:
- ☐ Certified copies of the priority documents have been received.
 - ☐ Certified copies of the priority documents have been received in Application No. _____.
 - ☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

- | | |
|---|---|
| 1) <input checked="" type="checkbox"/> Notice of References Cited (PTO-892) | 4) <input checked="" type="checkbox"/> Interview Summary (PTO-413) Paper No(s)/Mail Date. <u>8</u> . |
| 2) <input type="checkbox"/> Notice of Draftsperson's Patent Drawing Review (PTO-948) | 5) <input type="checkbox"/> Notice of Informal Patent Application (PTO-152) |
| 3) <input type="checkbox"/> Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date _____. | 6) <input type="checkbox"/> Other: _____. |

DETAILED ACTION

1. Claims 1-4, 6-13, and 20-22 have been examined

Papers Submitted

2. It is hereby acknowledged that the following papers have been received and placed of record in the file since the last correspondence:

#5: Amendment A (2/5/04)

#6: Amendment B (Supplemental) (2/20/04)

Specification

3. The specification is objected to as failing to provide proper antecedent basis for the claimed subject matter. See 37 CFR 1.75(d)(1) and MPEP § 608.01(o). Correction of the following is required: Claims 12 and 13 recite the limitation that a predetermined number of instructions are fetched depending upon the state of the branch instruction information. There is insufficient antecedent basis for this limitation in the claims.

Claim Rejections - 35 USC § 102

4. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

A person shall be entitled to a patent unless –

(b) the invention was patented or described in a printed publication in this or a foreign country or in public use or on sale in this country, more than one year prior to the date of application for patent in the United States.

5. Claims **21 and 22** rejected under 35 U.S.C. 102(b) as being anticipated by Chang (US005933628A).

6. **In regard to claim 21:**

7. Chang teaches a compiler, comprising:

compiler means for detecting a conditional branch instruction (col. 3, lines 50-51);

compiler means for determining the likelihood of successfully predicting the branch instruction (col. 3, lines 56-64); and

compiler means for embedding branch instruction information within the branch instruction, wherein the branch instruction information is indicative of the likelihood of successfully predicting the branch instruction (col. 6, lines 8-16).

8. **In regard to claim 22:**

9. Chang as applied to claim 21 further teaches the compiler of claim 21, wherein the means for embedding branch instruction information within the branch instruction (col. 6, lines 24-27) comprises means for embedding a first branch instruction information code (col. 6, lines 20-23) in the branch instruction if the branch instruction is

not likely to be predicted successfully and a second branch instruction code otherwise (hint bit is left unmarked; col. 6, lines 35-36).

Claim Rejections - 35 USC § 103

10. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

(a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negated by the manner in which the invention was made.

11. Claims **1-4, 6-13, and 20** are rejected under 35 U.S.C. 103(a) as being unpatentable over Sharangpani et al. (US005860017A) in view of Chang (US005933628A).

12. In regard to Claim 1:

13. Sharangpani et al. teach a method of executing instructions in a microprocessor (col. 16, line 47) comprising:

fetching a conditional branch instruction (col. 16, line 53) from an instruction cache (col. 5, lines 35-37);

responsive to branch prediction information indicating that the conditional branch instruction is not likely to be successfully predicted, fetching instructions from both a branch-taken path and from a branch-not-taken path of the branch instruction (col. 4, lines 6-16).

14. However, Shahrangpani et al. do not explicitly mention the step of detecting branch prediction information embedded in the branch instruction, wherein the embedded branch prediction information is indicative of the likelihood of successfully predicting the result of the conditional branch instruction.

15. Chang teaches a compiler that embeds a hint bit into a branch which indicates whether a branch is hard to predict or easy to predict (col. 3, lines 35-38; col. 6, lines 8-17, 21-23, 35-36). Depending on the hint bit, a different branch prediction mechanism is used (col. 6, lines 23-25; col. 6, lines 35-36, col. 4, lines 52-53). Although not explicitly mentioned, in order to select which branch prediction mechanism to use, a step of detecting the hint bit must be performed. Chang further teaches that the use of compile time heuristics to identify hard to predict branches can be used to optimize hardware space (col. 2, lines 41-46).

16. Therefore it would have been obvious to one of ordinary skill in the art at the time of the invention to have modified the invention of Shahrangpani et al. by using a compiler to embed the likelihood of successfully predicting the result of the conditional branch instruction in a hint bit in the instruction and detecting the hint bit in the

instruction instead of using the likelihood information set by hardware in the column indicating likeliness of correct prediction (Shahrangpani et al.: fig. 4, element 412).

17. One of ordinary skill in the art would have been motivated to do so because it would result in a simpler predictor and hence reduced hardware costs.

18. In regard to Claim 2:

19. Sharangpani et al. further teach of:

- 1) speculatively executing the instructions from the branch-taken path and the branch-not-taken path of the branch instruction (Fig. 6B, element 627, col. 14, lines 39-43);
- 2) executing (resolving, col. 17, line 37) the conditional branch instruction;
- 3) based upon the outcome of the conditional branch instruction, discarding the results from the speculatively executed instructions from the branch-taken path if the branch is not taken and discarding results from the branch-not-taken path if the branch is taken (col. 17, lines 38-41, 46-47).

20. In regard to Claim 3:

21. The combination of Sharangpani et al. in view of Chang as applied to claim 1 teaches the method of claim 1, wherein detecting the branch prediction information comprises detecting compiler generated branch prediction information.

22. In regard to Claim 4:

Art Unit: 2183

23. The combination of Sharangpani et al. in view of Chang as applied to claim 1 does not explicitly mention the method of claim 3, wherein the branch prediction information causes instruction fetching from both the taken and not taken branches if a compiler determines that the probability of successfully predicting the conditional branch instruction is less than approximately 75%.

24. However, the IBM technical disclosure bulletin document details a compiler that sets a H-bit in each branch instruction indicating whether or not the branch has random behavior. It discloses that when the percentage of a given conditional branch being taken or not taken is below 90%, the compiler sets the H-bit to 1, indicating that the branch has random behavior. It is understood that a branch having random behavior is difficult to predict.

25. Chang also mentions that the compiler may use a different set of heuristics to determine the predictability of a branch (col. 4, lines 18-19).

26. Therefore it would have been obvious to one of ordinary skill in the art at the time of the invention to modify the compiler to detect whether a branch is likely to be predicted correctly or not when the percentage of a given conditional branch being taken or not taken is below 90% (which is approximately 75%).

27. One would have been motivated to do so in order to simplify the compiler heuristics and reduce the amount of time to compile a code.

28. In regard to Claim 6:

Art Unit: 2183

29. Sharangpani et al. teach to fetch instructions from the branch-not-taken (sequential or second target) path (col. 4, lines 13-15). However they do not specifically mention to fetch down the branch-not-taken path until a subsequent branch instruction is encountered. It would have been obvious to one of ordinary skill in the art at the time of the invention to fetch down the branch-not-taken path until a subsequent branch instruction is encountered because this follows normal instruction fetch semantics.

30. In regard to Claim 7:

31. Shahrangpani et al. teach a microprocessor (Fig. 3, element 101) comprising:
an instruction cache (Fig. 3, element 301) suitable for storing a set of processor executable instructions and configured to receive an instruction address and to retrieve an instruction corresponding to the instruction address; and

a fetch unit (Fig. 3, element 304) connected to the instruction cache and configured to generate an instruction address;

32. Although Shahrangpani et al. teach of fetching from both a branch-taken path and a branch-not-taken path of the branch instruction if the branch prediction information indicates that the branch instruction is not likely to be predicted successfully (col. 4, lines 6-16), they do not teach that the fetch unit is configured to detect branch instruction information embedded in a branch instruction retrieved from the instruction cache.

33. Chang teaches a compiler that embeds a hint bit into a branch which indicates whether a branch is hard to predict or easy to predict (col. 3, lines 35-38; col. 6, lines 8-

Art Unit: 2183

17, 21-23, 35-36). Depending on the hint bit, a different branch prediction mechanism is used (col. 6, lines 23-25; col. 6, lines 35-36, col. 4, lines 52-53). Although not explicitly mentioned, in order to select which branch prediction mechanism to use, a step of detecting the hint bit must be performed. Chang further teaches that the use of compile time heuristics to identify hard to predict branches can be used to optimize hardware space (col. 2, lines 41-46).

34. Therefore it would have been obvious to one of ordinary skill in the art at the time of the invention to have modified the invention of Shahrangpani et al. by using a compiler to embed the likelihood of successfully predicting the result of the conditional branch instruction in a hint bit in the instruction and enabling the fetch unit to detect the hint bit in the instruction instead of using the likelihood information set by hardware in the column indicating likeliness of correct prediction (Shahrangpani et al.: fig. 4, element 412).

35. One of ordinary skill in the art would have been motivated to do so because it would result in a simpler predictor and hence reduced hardware costs.

36. In regard to Claim 8:

37. The combination of Sharangpani et al. in view of Chang as applied to claim 7 teaches the microprocessor of claim 7, further comprising a branch prediction unit (fig. 4, 336) comprising prediction logic invoked by the branch instruction information only if the embedded branch prediction information indicates that the branch instruction is likely to be predicted successfully (Sharagpani et al.: fig. 6A shows that if the branch is

likely to be predicted successfully [612], then prediction logic is used to fetch instructions [615]; As per the modifications made by the combination of Shahrangpani and Chang, the step 612 of determining whether the branch is likely to be predicted successfully or not is done by detecting the state of the hint bit embedded in the instruction set by the compiler) and configured to predict the result of a branch instruction based on information in a branch history table (fig. 4, elements 408,410).

38. In regard to Claim 9:

39. The combination of Sharangpani et al. in view of Chang as applied to claim 7 teaches the microprocessor of claim 8, wherein the branch prediction unit includes a prediction bypass unit (Shahrangpani: fetch unit, Fig. 3, element 304) invoked only if the branch prediction information indicates that the branch instruction is not likely to be predicted successfully (Shahrangpani: col. 9, lines 6-10) and configured to issue instruction addresses (Shahrangpani: fetches, col. 9, line 12) from a branch-taken path and a branch-not-taken path of the branch instruction (Shahrangpani: col. 9, lines 10-13) (As per the modifications made by the combination of Shahrangpani and Chang, the step 612 in fig 6A of determining whether the branch is likely to be predicted successfully or not, is done by detecting the state of the hint bit embedded in the instruction set by the compiler. This bit would therefore invoke the bypass unit if set to 1 i.e. branch is unlikely to be predicted accurately).

40. In regard to Claim 10:

41. The combination of Sharangpani et al. in view of Chang as applied to claim 7 teaches the microprocessor of claim 7, wherein

the processor is configured to speculatively execute the instructions from the branch-taken path and from the branch-not-taken path of the branch instruction (Fig. 6B, element 627, col. 14, lines 39-43),

execute (resolving, col. 17, line 37) the conditional branch instruction, and, based upon the outcome of the conditional branch instruction, discard results from the speculatively executed instructions from the branch-taken path if the branch is not taken and discarding results from the branch-not-taken path if the branch is taken (col. 17, lines 38-41, 46-47).

42. In regard Claim 11:

43. The combination of Sharangpani et al. in view of Chang as applied to claim 7 teaches the microprocessor of claim 7 wherein the branch prediction information comprises compiler generated information.

44. In regard to Claim 12:

45. Sharangpani et al. teach to fetch instructions from the branch-taken (target) and branch-not-taken (sequential or second target) paths (col. 4, lines 13-15) depending upon the state of the branch instruction information (branch prediction information) but do not explicitly mention to fetch the same predetermined number of instructions from both paths. However, it is inherent to fetch a predetermined number of instructions

because the processor has to follow specific instruction fetch semantics. Moreover, if both the paths were to be fetched, there will be a point in time when at least one instruction is fetched from both paths.

46. In regard to Claim 13:

47. Sharangpani et al. teach to fetch instructions from the branch-not-taken (sequential or second target) path (col. 4, lines 13-15) depending upon the state of the branch instruction information (branch prediction information). However they do not specifically mention to fetch down the branch-not-taken path until a subsequent branch instruction is encountered. However, this is deemed inherent because according to normal fetching semantics, fetching down any path would continue until it is not known where to fetch from i.e. a conditional branch is encountered.

48. In regard to claim 20:

49. The method of claim 4, further comprising, responsive to the branch prediction information (Chang: hint bit) indicating that the conditional branch instruction is likely to be successfully predicted, using a branch history table (Shahrangpani: fig. 4, elements 408, 410) to predict the branch and, based upon the prediction, fetching instructions from only the predicted path (Sharagpani et al.: fig. 6A shows that if the branch is likely to be predicted successfully [612], then prediction logic is used to fetch instructions from the predicted path only [615]; As per the modifications made by the combination of Shahrangpani and Chang, the step 612 of determining whether the branch is likely to be

predicted successfully or not is done responsive to the state of the hint bit embedded in the instruction set by the compiler).

Response to Arguments

Applicant's arguments with respect to claims **1, 4, 7, 8, 21, and 22** have been considered but are moot in view of the new ground(s) of rejection.

Conclusion

50. Applicant's amendment necessitated the new ground(s) of rejection presented in this Office action. Accordingly, **THIS ACTION IS MADE FINAL**. See MPEP § 706.07(a). Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire **THREE MONTHS** from the mailing date of this action. In the event a first reply is filed within **TWO MONTHS** of the mailing date of this final action and the advisory action is not mailed until after the end of the **THREE-MONTH** shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than **SIX MONTHS** from the date of this final action.

Art Unit: 2183

51. The prior art made of record and not relied upon is considered pertinent to applicant's disclosure. Applicant is reminded that in amending in reply to a rejection of claims in an application or patent under reexamination, the applicant or patent owner must clearly point out the patentable novelty, which he or she thinks the claims present in view of the state of the art disclosed by the references cited or the objections made. The applicant or patent owner must also show how the amendments avoid such references or objections. See 37 CFR § 1.111.

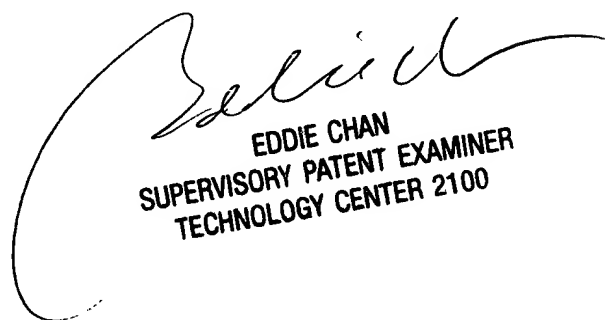
- a. A. Unger et al. ("A Combined Compiler and Architecture Technique to Control Multithreaded Execution of Branches and Loop Iterations", SIGARCH Comput. Archit. News, vol. 28, no.1, March 2000, pp. 53-61) teach a unpredictable branches and dual path execution.
- b. Soummya et al. (EP000805390A1) teach a branch prediction mechanism that uses both static and dynamic prediction methods.
- c. S. Mahlke and B. Natarajan ("Compiler Synthesized Dynamic Branch Prediction," in Proceedings of the 29th Annual ACM/IEEE International Symposium and Workshop on Microarchitecture, pp. 153--164, December 1996.) teach of a compiler that can do branch prediction and which inserts branch prediction instructions after the branch instruction.
- d. Dhong et al. (US006334184B1) teach static predictions based on bits associated with the instruction (col. 5, lines 7-10).

52. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Amol V. Gole whose telephone number is 703-305-8888. The examiner can normally be reached on 9:00-5:30.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Eddie Chan can be reached on 703-305-9712. The fax phone number for the organization where this application or proceeding is assigned is 703-872-9306.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see <http://pair-direct.uspto.gov>. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

AVG
amol.gole@uspto.gov



EDDIE CHAN
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2100